(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 102625183 B
(45)授权公告日 2016.01.13

(21)申请号 201210104203.6
(22)申请日 2012.04.10
(73)专利权人 北京邮电大学
地址 100876 北京市海淀区西土城路10号
(72)发明人 双锴 覃国旗 徐鹏 王玉龙 于晓燕 苏森
(74)专利代理机构 北京思创毕升专利事务所 11218
代理人 刘明华
(51)Int.Cl.
H04N 21/433(2011.01)
H04N 21/438(2011.01)

(56)对比文件
CN 102281290 A,2011.12.14, 全文.
WO 00/27106 A2,2000.05.11, 全文.

审查员 王从雷

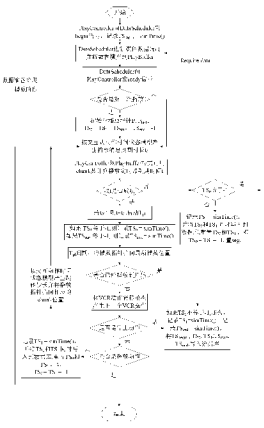
权利要求书3页 说明书8页 附图3页

(54)发明名称

一种用于点播系统仿真的用户终端播放方法

(57)摘要

本发明涉及一种用于点播系统仿真的用户终端播放方法,包括以下步骤:(1)先进行模型的参数配置,(1.1)将VCR动作转移模型的概率采用通行假设,即设定Pr_{play},Pr_{abort},Pr_{jf},Pr_{jb}的值,并且满足$Pr_{play}+Pr_{abort}+Pr_{jf}+Pr_{jb}=1$;(1.2)将VCR动作状态模型中的参数采用通行数据,即设定播放时间的参数,跳转持续时间的参数;(2)整个节目视频播放过程可以分为数据准备和播放两个阶段;(3)性能指标计算,仿真结束,模型采集到的性能指标的原始数据用于计算两个重要的用户侧性能指标,分别是节目启动时延和节目播放连续度,仿真过程所采集的原始数据均存储于数据库之中,可通过数据库平台进行计算。这两个指标主要体现用户的观看体验,使用户得到更好的仿真观看体验。



CN 102625183 B

1.一种用于点播系统仿真的用户终端播放方法,其特征在于,包括以下步骤:

(1) 先进行模型的参数配置,

(1.1) 将 VCR 动作转移模型的概率采用通行假设,即设定表示上一次播放动作结束之后,下一个动作是播放 (Play) 的概率 Pr_{play}的值;表示上一次播放动作结束之后,下一个动作是停止 (Abort) 的概率 Pr_{abort}的值;表示上一次播放动作结束之后,下一个动作是跳进 (JF) 的概率 Pr_{jf}的值;表示上一次播放动作结束之后,下一个动作是跳退 (JB) 的概率 Pr_{jb} 的值,并且满足 $Pr_{play}+Pr_{abort}+Pr_{jf}+Pr_{jb}=1$;

(1.2) 将 VCR 动作状态模型中的参数采用通行数据,即设定播放时间的参数,跳转持续时间的参数;

(2) 整个节目视频播放过程分为数据准备和播放两个阶段;

所述的步骤 (2) 进一步包括:

(2.1) 数据准备阶段,

步骤 2.1.1,播放控制模块 PlayController 向数据调度模块 DataScheduler 发送一个开始 begin 消息置 seg = 0,表示第一块媒体数据,数据调度模块 DataScheduler 开始进行数据调度,播放控制模块 PlayController 记录时间戳 TS_{begin},即 TS_{begin}= simTime0,其中 simTime0 表示取当前仿真系统的时间戳;

步骤 2.1.2,数据调度模块 DataScheduler 进行数据调度对 seg 进行调度,把收满的 BM 数据块转移到播放缓冲区 PlayBuffer,同时为了最大保证播放的连续,会用自身的数据调度算法进行其它数据的调度;转到步骤 2.1.3;

步骤 2.1.3,数据调度模块 DataScheduler 向播放控制模块 PlayController 发送 ready 信号,告知播放控制模块 PlayController 已经可以开始进行播放;此时,进入播放阶段;

(2.2) 播放阶段,

步骤 2.2.1,是否是第一次开始播放,若是,则转到步骤 2.2.2,否则转到步骤 2.2.3;

步骤 2.2.2,初始化 P, P_{last}, TS_0, TS_1, TS_{start}, TS_{end}使得 P.offset = 0, P.chunk = 0, P_{last}.offset = 0, P_{last}.chunk = 0, TS_0= -1, TS_1= -1, TS_{start}= -1, TS_{end}= -1,即让 P 和 P_{last}都指向播放缓冲区 PlayBuffer 的起始位置;

P 的含义是播放指针,表示当前的播放位置;

P_{last}的含义是记录上次播放位置;

Ts_{start}的含义是进行第一次播放动作的仿真时间戳;

TS_0的含义是进行一次播放动作开始的仿真时间戳;

TS_1的含义是一次播放动作结束的仿真时间戳;

TS_{end}的含义是用户会话结束的仿真时间戳;

步骤 2.2.3,由公式分布密度函数是 $f(x)=\frac{e^{-(\ln x-\mu)^2}}{x\sigma\sqrt{2\pi}},x>0$ 产生播放时间步长的随机数 t_0;

步骤 2.2.4,播放控制模块 PlayController 从播放缓冲区 PlayBuffer 根据可用 BM 个数计算播放定时器的持续时间 t,方法是:从 P 点处往后遍历播放缓冲区 PlayBuffer,直到 BM 的 offset 不等于 -1,记录 BM 的个数 n,若 n > 0,那么计算 n 块 BM 可以播放的时间 t,计

算公式：t = n*(BMSize/BitRate)，跳到步骤 2.2.8；若 n ≤ 0，转到步骤 2.2.5；

步骤 2.2.5，判断若 TS_0 等于 -1，则跳到步骤 2.2.7；否则跳到步骤 2.2.6；

步骤 2.2.6，记录 TS_1 = simTime0，将 TS_0，TS_1 同时写入数据库，并重置 TS_0，TS_1，即 TS_0 = -1，TS_1 = -1；

步骤 2.2.7，那么播放控制模块 PlayController 向数据调度模块 DataScheduler 发送请求数据 require data 的信号，信号中携带有 seg 信息，表示出现“卡”的位置的数据块，转到数据准备阶段的步骤 2.1.2；

步骤 2.2.8，若 t_0 > t，那么令 t = t_0；

步骤 2.2.9，启动定时器 T_{p1}，其到期时间间隔为 t. 若 TS_0 等于 -1，则记录 TS_0 = simTime0，若 TS_{start} 等于 -1，则记录 TS_{start} = simTime0；

步骤 2.2.10，定时器 T_{p1} 到期，把指针 P 指向当前播放处，计算公式是 P.offset = P_{last}.offset+n；

播放过的时间记录到 P_{last} 中，计算公式是 P_{last}.offset = P_{last}.offset+n；

步骤 2.2.11，检查是否已经播放到片尾，即 P.offset 是否等于 N，且 P.chunk 是否等于 M_{last}，若是，则跳到步骤 2.2.17，否则跳到步骤 2.2.12；

M_{last} 表示最后一块 BM 包含的 chunk 个数，其计算方法为：

M_{last} = MediaSizemodChunkSize，其中 ChunkSize 表示一个 Chunk 数据块大小，mod 是模运算；

N 的含义是 [MediaSize/BMSize]，其中 MediaSize 表示媒体视频文件的大小，单位是 byte；BMSize 表示 BM 数据块大小，单位是 byte；

步骤 2.2.12，按照 VCR 动作转移模型产生下一个 VCR 操作；

步骤 2.2.13，该 VCR 操作是否为停止，若是，则跳到步骤 2.2.17；否则跳到步骤 2.2.14；

步骤 2.2.14，判断该 VCR 操作是否为播放，若是，则跳到步骤 2.2.3；否则跳到步骤 2.2.15；

步骤 2.2.15，对跳进和跳退操作，由公式分布密度函数采用参数产生跳转的步长 t_{step}，其包含的 chunk 个数为 n_{chunk} = t_{step}/(ChunkSize/BitRate)，包含的 BM 个数为 n_{BM} = n_{chunk}/M，多余的 chunk 个数 r_{chunk} = n_{chunk} mod M，其中 M 表示每个 BM 包含 chunk 的个数；将播放指针重置，若是跳进动作，则 P.offset = P.offset+n_{BM}，P.chunk = P.chunk+r_{chunk}；若是跳退动作，则 P.offset = P.offset-n_{BM}，P.chunk = P.chunk-r_{chunk}，转到步骤 2.2.16；

步骤 2.2.16，记录 TS_1 = simTime0，将 TS_0，TS_1 同时写入数据库，并重置 TS_0，TS_1，即 TS_0 = -1，TS_1 = -1，转到步骤 2.2.3；

步骤 2.2.17，若 TS_0 不等于 -1，则记录 TS_1 = simTime0，记录 TS_{end} = simTime0，将 TS_{begin}，TS_0，TS_1，TS_{start}，TS_{end} 同时写入数据库；

步骤 2.2.18，播放结束；若定时器 T_{p1} 仍处在调度状态，则将之取消；

(3) 性能指标计算，仿真结束，模型采集到的性能指标的原始数据用于计算两个重要的用户侧性能指标，分别是节目启动时延和节目播放连续度，仿真过程所采集的原始数据均存储于数据库之中，可通过数据库平台进行计算；

(1) 节目启动时延：T_{start} = TS_{start}-TS_{begin}，

(2) 节目播放连续度：由定义可知，C_{play}= t_{play}/t_{all}，在仿真过程中，假设有 n 对数据对 <TS_0, TS_1> 存储入数据库当中，则 $t_{play}=\sum_{i=1}^{n}(TS_{ai}-TS_{1i})$，而 t_{all}= TS_{start}-TS_{end}，故有 $C_{play}=\sum_{i=1}^{n}(TS_{ai}-TS_{1i})/(TS_{start}-TS_{end})$。

2. 根据权利要求 1 所述的用于点播系统仿真的用户终端播放方法，其特征在于，所述的步骤 2.2.3 中采用参数 μ = -4.19，σ = 1.13。

3. 根据权利要求 1 所述的用于点播系统仿真的用户终端播放方法，其特征在于，所述的步骤 2.2.15 中采用参数 μ = -3.76，σ = 1.01。

4. 根据权利要求 1 所述的用于点播系统仿真的用户终端播放方法，其特征在于，所述的步骤 2.1.2 中 BM 数据块为，把一个媒体视频文件被划分为 N(N > 0) 个固定大小的 Buffer Message 块，简称为 BM 数据块，每个 BM 又被分为 M(M > 0) 份，每一份称为一个 chunk，用数字“0”和“1”来表示该 chunk 数据存在与否，如果该位置上是“1”，则表示有该 chunk 数据块，“0”则表示没有。

一种用于点播系统仿真的用户终端播放方法

技术领域

[0001] 本发明涉及基于数学计算的多媒体播放仿真方法，具体地涉及属于计算机仿真领域的多媒体播放器仿真技术领域。

背景技术

[0002] 计算机仿真是借助高速、大存储量数字计算机及相关技术，对复杂真实系统的运行过程或状态进行数字化模拟的技术。在科技发展的今天，对复杂真实系统的研究代价昂贵且难以实现，因此有引进计算机仿真的必要。

[0003] 近年来随着互联网的发展以及宽带应用的普及，流媒体业务成为了网络应用中的“杀手级”业务，比如直播、点播等业务。研究人员在研究流媒体系统时往往会关注能体现系统性能的一些主要参数，比如节目播放启动时延，节目播放连续度，流媒体系统服务器压力，网络端到端传输时延等。然而对这些性能参数的测量一般需要参考系统中所有节点的运行情况进行监控，通过对监控数据进行分析得出测试结果。由于要在实际网络环境下对系统进行测试是比较困难，而且成本昂贵，在这种情况下，计算机仿真成为有效的解决手段。

[0004] 对流媒体点播系统进行研究时，用户对某些媒体内容的喜爱偏好所发生的用户交互行为对数据调度效率的影响，往往成为研究人员需要考虑的方面。用户的交互行为集中体现在常见的VCR(Video Cassette Recorder)操作上，比如跳进，跳退，播放，停止等，这些操作对媒体系统的性能存在着潜在的影响。

[0005] 在流媒体点播系统仿真中，媒体播放模型主要用于模拟仿真用户的这些VCR操作行为，为研究者人员提供从用户行为的角度考量系统的设计，从而提高媒体数据调度效率，优化网络资源使用率，改善用户体验。研究者可以从该模型中采集能够体现用户体验的两个重要的性能参数：节目启动时延和节目播放连续度。

发明内容

[0006] 本发明的目的是提供一种用于仿真的流媒体点播系统的播放方法，该方法可仿真固定码率(CBR)流媒体的播放行为，给研究者提供基于用户喜爱偏好的VCR操作的点播系统的各个性能参数，对这些参数进行分析可以为研究者设计、优化点播系统提供参考。该模型运用数学计算和仿真平台的自消息机制相结合的方式，控制播放指针的变动，实现常见的VCR操作，这些操作包括播放、停止、跳进、跳退等。

[0007] 一种用于点播系统仿真的用户终端播放方法，包括以下步骤：

[0008] (1)先进行模型的参数配置，

[0009] (1.1)将VCR动作转移模型的概率采用通行假设，即设定表示上一次播放动作结束之后，下一个动作是播放(Play)的概率Pr_{play}的值；表示上一次播放动作结束之后，下一个动作是停止(Abort)的概率Pr_{abort}的值；表示上一次播放动作结束之后，下一个动作是跳进(JF)的概率Pr_{jf}的值；表示上一次播放动作结束之后，下一个动作是跳退(JB)的概率

Pr_{jb}的值，并且满足 $Pr_{play}+Pr_{abort}+Pr_{jf}+Pr_{jb}=1$；

[0010] (1.2) 将 VCR 动作状态模型中的参数采用通行数据，即设定播放时间的参数，跳转持续时间的参数；

[0011] (2) 整个节目视频播放过程可以分为数据准备和播放两个阶段；

[0012] (3) 性能指标计算，仿真结束，模型采集到的性能指标的原始数据用于计算两个重要的用户侧性能指标，分别是节目启动时延和节目播放连续度，仿真过程所采集的原始数据均存储于数据库之中，可通过数据库平台进行计算。

[0013] 所述的步骤 (1.1) 中设定 $Pr_{play}=0.70$，$Pr_{abort}=0.05$，$Pr_{jf}=0.15$，$Pr_{jb}=0.10$。

[0014] 所述的步骤 (1.2) 中播放时间的参数是 $\mu=-4.19$，$\sigma=1.13$，跳转持续时间的参数是 $\mu=-3.76$，$\sigma=1.01$。

[0015] 所述的步骤 (2) 进一步包括：

[0016] (2.1) 数据准备阶段，

[0017] 步骤 2.1.1 播放控制模块 PlayController 模块向数据调度模块 DataScheduler 发送一个 begin 消息置 seg = 0，表示第一块媒体数据，数据调度模块 DataScheduler 开始进行数据调度，播放控制模块 PlayController 记录时间戳 TS_{begin}，即 TS_{begin} = simTime0，其中 simTime0 表示取当前仿真系统的时间戳；

[0018] 步骤 2.1.2 数据调度模块 DataScheduler 进行数据调度对 seg 进行调度，把收满的 BM 数据块转移到播放缓冲区 PlayBuffer，同时为了最大保证播放的连续，会用自身的数据调度算法进行其它数据的调度；转到步骤 2.1.3；

[0019] 步骤 3 数据调度模块 DataScheduler 向播放控制模块 PlayController 发送 ready 信号，告知播放控制模块 PlayController 已经可以开始进行播放；此时，进入播放阶段。

[0020] 所述的步骤 (2) 进一步包括：

[0021] (2.2) 播放阶段，

[0022] 步骤 2.2.1 是否是第一次开始播放，若是，则转到步骤 2.2.2，否则转到步骤 2.2.3；

[0023] 步骤 2.2.2 初始化 P，P_{last}，TS_0，TS_1，TS_{start}，TS_{end}使得 P.offset = 0，P.chunk = 0，P_{last}.offset = 0，P_{last}.chunk = 0，$TS_0=-1$，$TS_1=-1$，$TS_{start}=-1$，$TS_{end}=-1$，即让 P 和 P_{last}都指向播放缓冲区 PlayBuffer 的起始位置；

[0024] 步骤 2.2.3 由公式分布密度函数是$f(x)=\frac{e^{-(\ln x-\mu)^2}}{x\sigma\sqrt{2\pi}},x>0$产生播放时间步长的随机数 t_0；

[0025] 步骤 2.2.4 播放控制模块 PlayController 从播放缓冲区 PlayBuffer 根据可用 BM 个数计算播放定时器的持续时间 t，方法是：从 P 点处往后遍历播放缓冲区 PlayBuffer，直到 BM 的 offset 不等于 -1，记录 BM 的个数 n. 若 n > 0，那么计算 n 块 BM 可以播放的时间 t，计算公式：t = n • (BMSize/BitRate)，跳到步骤 2.2.8；若 n ≤ 0，转到步骤 2.2.5；

[0026] 步骤 2.2.5 判断若 TS_0等于 -1，则跳到步骤 2.2.7；否则跳到步骤 2.2.6；

[0027] 步骤 2.2.6 记录 TS_1 = simTime0，将 TS_0，TS_1同时写入数据库，并重置 TS_0，TS_1，即 $TS_0=-1$，$TS_1=-1$；

[0028] 步骤 2.2.7 那么播放控制模块 PlayController 向数据调度模块 DataScheduler 发送 require data 的信号，信号中携带有 seg 信息，表示出现“卡”的位置的数据块，转到

数据准备阶段的步骤 2.1.2；

[0029] 步骤 2.2.8 若 $t_0 > t$，那么令 $t = t_0$；

[0030] 步骤 2.2.9 启动定时器 T_{pt}，其到期时间间隔为 t，若 TS_0 等于 -1，则记录 TS_0 = simTime0，若 TS_{start} 等于 -1，则记录 TS_{start} = simTime0；

[0031] 步骤 2.2.10 定时器 T_{pt} 到期，把指针 P 指向当前播放处，计算公式是

[0032] P.offset = P_{last}.offset+n；

[0033] 播放过的时间记录到 P_{last} 中，计算公式是

[0034] P_{last}.offset = P_{last}.offset+n；

[0035] 步骤 2.2.11 检查是否已经播放到片尾，即 P.offset 是否等于 N，且 P.chunk 是否等于 M_{last}，若是，则跳到步骤 2.2.17，否则跳到步骤 2.2.12；

[0036] 步骤 2.2.12 按照 VCR 动作转移模型产生下一个次 VCR 操作；

[0037] 步骤 2.2.13 该 VCR 操作是否为停止，若是，则跳到步骤 2.2.17；否则跳到步骤 2.2.14；

[0038] 步骤 2.2.14 判断该 VCR 操作是否播放，若是，则跳到步骤 2.2.3；否则跳到步骤 2.2.15；

[0039] 步骤 2.2.15 对跳进和跳退操作，由公式分布密度函数是产生跳转的步长 t_{step}，其包的 chunk 个数为 $n_{chunk} = t_{step}/(ChunkSize/BitRate)$，包含的 BM 个数为 $n_{BM} = n_{chunk}/M$，多余的 chunk 个数 $r_{chunk} = n_{chunk} \bmod M$，其中 M 表示每个 BM 包含 chunk 的个数；将播放指针重置，若是跳进动作，则 P.offset = P.offset+n_{BM}，P.chunk = P.chunk+r_{chunk}；若是跳退动作，则 P.offset = P.offset-n_{BM}，P.chunk = P.chunk-r_{chunk}，转到步骤 2.2.16；

[0040] 步骤 2.2.16 记录 TS_1 = simTime0，将 TS_0，TS_1 同时写入数据库，并重置 TS_0，TS_1，即 TS_0 = -1，TS_1 = -1，转到步骤 2.2.3；

[0041] 步骤 2.2.17 若 TS_0 不等于 -1，则记录 TS_1 = simTime0，记录 TS_{end} = simTime0，将 TS_{begin}，TS_0，TS_1，TS_{start}，TS_{end} 同时写入数据库；

[0042] 步骤 2.2.18 播放结束；若定时器 T_{pt} 仍处在调度状态，则将之取消。

[0043] 仿真过程中，该模型可采集用户 VCR 操作的原始数据，在仿真结束后，研究者可通过这些参数计算出两个重要用户侧的性能指标，分别是节目启动时延和节目播放连续度，这两个指标主要体现用户的观看体验。使用户得到更好的仿真观看体验。

附图说明

[0044] 图 1 是媒体数据分块。

[0045] 图 2 是 VCR 动作转移示意图。

[0046] 图 3 是交互式动作时间。

[0047] 图 4 是用户终端结构图。

[0048] 图 5 是播放缓冲区示意图。

[0049] 图 6 是视频播放步骤流程图。

具体实施方式

[0050] 下面结合实施例进一步描述本发明。本发明的范围不受这些实施例的限制，本发

明的范围在权利要求书中提出。

[0051] 首先，对流媒体相关概念背景进行介绍，然后对交互式用户行为模型进行分析，最后对播放模型中出现的变量进行定义，详细阐述专利的用户终端播放模型，并给出运用原始数据计算节目启动时延和节目播放连续度的方法。

[0052] 一，相关背景定义：

[0053] 1，媒体数据分块定义

[0054] 文献《基于网络测量的 PPStream 网络电视系统研究》，作者：李代玲，硕士论文 .2008.6 页 5-6，中把一个媒体视频文件被划分为 N(N ＞ 0) 个固定大小的 Buffer Message 块，简称为 BM 数据块。每个 BM 又被分为 M(M ＞ 0) 份，每一份称为一个 chunk，用数字“0”和“1”来表示该 chunk 数据存在与否，如果该位置上是“1”，则表示有该 chunk 块，“0”则表示没有，如图 1 所示。Offset 是 BM 的唯一标识，表示 BM 与视频段的初始位置的偏移量，用一个无符号的整型数值表示。因为每个 Offset 对应一个 BM，所以可以用 Offset 的值来代指 BM 的位置。

[0055] 那么，一个媒体视频文件所包含的 BM 的个数的计算方法见 (1) 式：

[0056] $N = [MediaSize/BMSize]$ (1)

[0057] 其中 MediaSize 表示媒体视频文件大小，单位是 byte；BMSize 表示 BM 数据块大小，单位是 byte；$offset_i$ ($i = 0,1,2...N-1$) 表示第 i 块 BM 数据块。每个 BM 所包含的 chunk 个数的计算方法见 (2) 式：

[0058] $M = BMSize/ChunkSize$ (2)

[0059] 其中 BMSize 表示 BM 数据块大小，单位为 byte；ChunkSize 表示一个 chunk 数据块的大小，单位为 byte.

[0060] 由于 MediaSize 并不一定 BMSize 的整数倍，所以最后一个 BM 数据块小于或者等于前面其它数据块的大小，对最后一个 BM 数据块进行处理，M_{last} 表示最后一块 BM 包含的 chunk 个数，其计算方法见 (3) 式：

[0061] $M_{last} = MediaSize \bmod ChunkSize$ (3)

[0062] 其中 ChunkSize 表示一个 chunk 数据块大小，mod 是模运算。

[0063] 2，交互式用户行为模型

[0064] 2.1，VCR 动作转移模型

[0065] 本发明只考虑常用的交互式动作，这些动作包括播放 (Play)、暂停 (Pause)、跳进 (JF，Jump Forward)、跳退 (JB，Jump Back)、停止 (Abort)。

[0066] VCR 动作转移如图 2 所示，每个动作只依赖于前一个动作类型，且用户必须在上个动作结束之后才能产生新的动作，动作的类型仅仅依赖于前一个动作。下面首先定义几个概念：

[0067] (1) Pr_{play} 表示上一次播放动作结束之后，下一个动作是播放 (Play) 的概率。

[0068] (2) Pr_{abort} 表示上一次播放动作结束之后，下一个动作是停止 (Abort) 的概率。

[0069] (3) Pr_{jf} 表示上一次播放动作结束之后，下一个动作是跳进 (JF) 的概率。

[0070] (4) Pr_{jb} 表示上一次播放动作结束之后，下一个动作是跳退 (JB) 的概率。

[0071] 并且满足 $Pr_{play}+Pr_{abort}+Pr_{jf}+Pr_{jb} = 1$.

[0072] 在图 2 中可以看到，当媒体数据准备好之后，此时进入播放状态 (Play)，播放动作

结束之后，分别以相应的概率跳转到下一个动作。并且假定跳进和跳退动作之后转到播放的转移概率为1，即在跳进动作（JF）和跳退动作（JB）之后都转移到播放动作（Play）；当状态转移到停止，整个播放过程结束。

[0073] 2.2，VCR动作时间状态模型

[0074] 将视频对象等分为可数的播放单元，如图3所示。播放单元是VCR操作的原子级对象，其对应着一定播放时间的存储单元，例如一个chunk大小的数据。用户会话过程可被描述为播放单元的状态转移过程。在某个播放单元上所采用某个VCR动作的概率是上个动作结束后，由VCR动作转移模型描述；VCR动作所持续的时间由VCR动作时间状态模型描述，如图3中所示的播放时间t，跳进步长 t_{step} 和跳退步长 t'_{step}. 文献《A client caching scheme for interactive video-on-demand.（一个适用于交互式点播系统的客户端缓存方案）》In Proc. IEEE of Intentional conf. on Networks(ICON)1999, Melbourne, Australia, September 1999:391-397. 作者Branch P. Egan G. Tonkin B和文献《Continuous-media courseware server:A study of client interactions.（连续媒体课件服务器：一项用户交互的研究）》IEEE Internet Computing, Magagine 1999,3(6):65-73. 作者Padhye J. Kurose J研究指出，播放和跳转动作的持续时间呈现对数正态分布，分布密度函数是

[0075]

$$f(x)=\frac{e^{-(\ln x-\mu)^2}}{x\sigma\sqrt{2\pi}},x>0 \qquad (4)$$

[0076] 图3所示是交互式动作时间状态模型的生成过程：首先，当媒体数据准备好之后，用户第一个动作为正常播放，播放距离是服从（4）式的随机数，若该随机数大小超过目前缓冲区内数据所能播放的最长距离，那么播放距离由后者决定；在此播放距离播放结束后，由VCR动作转移模型选择下一个动作，播放或者跳转的距离为服从（4）式的随机数。此过程不断重复进行下去，直到用户会话的结束，会话结束的标志是发生停止操作或者正常播放结束，播放指针到达媒体结尾。

[0077] 3，相关变量定义

[0078] 定义1BitRate表示流媒体视频的码率，单位为bps。

[0079] 定义2 T_{pt} 表示播放定时器，t表示定时器的持续时间。

[0080] 定义3P是指播放指针，表示当前的播放位置，它是一个二元组 $P<offset_i, ChunkID_j>$ 其中 $0 \leqslant i < N, 0 \leqslant j < M$，表示当前播放的位置是第offset块BM数据块中的第ChunkID块chunk，其中ChunkID表示chunk的编号。

[0081] 定义4 P_{last} 记录上次播放位置，与P是同一类型的二元组，记录上次播放的基本信息。

[0082] 定义5 TS_{begin} 表示开始进行第一次数据调度的仿真时间戳。

[0083] 定义6 TS_0 表示进行一次播放动作开始的仿真时间戳，与 TS_1 之差，称为一次播放动作的时间步长。

[0084] 定义7 TS_1 表示是一次播放动作结束的仿真时间戳，与 TS_0 之差，称为一次播放动作的时间步长。

[0085] 定义8 TS_{start} 表示进行第一次播放动作的仿真时间戳。

[0086] 定义 $9TS_{end}$表示用户会话结束的仿真时间戳，即发生停止 (Abort) 动作或者正常播放结束。

[0087] 定义 $10T_{start}$表示节目播放启动时延，从用户开始申请收看视频节目时刻到发生第一次播放动作的时刻，这段时间差称为节目播放启动时延，用 T_{start}表示，单位为秒。由定义 5 和定义 8 可知，节目播放启动时延可表示为 $T_{start} = TS_{start} - TS_{begin}$。

[0088] 定义 $12C_{play}$表示节目播放连续度，指的是节目连续播放的时间与整个节目观看过程全部时间之比，假设节目连续播放的时间总和为 t_{play}，整个节目观看过程全部时间为 t_{all}，那么，节目播放连续度定义为 $C_{play} = t_{play}/t_{all}$。

[0089] 二，原理以及技术实现

[0090] 1，模型描述

[0091] 图 4 所示的是用于点播系统仿真的一个用户终端结构图，由三个部分组成，分别是 DataScheduler，PlayBuffer 和 PlayController. 下面分别对这三个部分的功能进行介绍。DataScheduler，数据调度模块，主要负责对媒体数据进行调度，其采用何种调度算法在发明中不做限制，它负责的工作主要有三部分 ：

[0092] (1) 接收来自 PlayController 的 begin 指令，开始进行第一次数据调度 ；

[0093] (2) 接收来自 PlayController 的调度某个数据块 seg 的指令，负责进行调度媒体数据调度 ；

[0094] (3) 接收数据的数据块，将之转移到数据缓冲区 PlayBuffer 中，并通知 PlayController 媒体数据块已经准备好。

[0095] PlayBuffer，播放缓冲区，主要是用于存储从 DataScheduler 传送过来的媒体数据块。其结构如图 5 所示，P 即为播放器播放的当前位置。缓冲区是由一系列的“1”和“0”，其中“1”代表着当前的 chunk 数据已经下载完成，“0”代表没有下载完成。

[0096] PlayController，播放控制模块的主要功能有 ：

[0097] (1) 向 DataScheduler 发送 begin 信号，告知开始进行数据调度。

[0098] (2) 产生用户 VCR 操作，规则遵循 VCR 动作转移模型和 VCR 时间状态模型，通过公式计算和播放定时器控制播放指针的变动，以此来模拟用户实际操作行为。

[0099] (3) 向 DataScheduler 发送调度某块紧急数据的请求，该数据块用 seg 表示，表示的是 BM 数据块的 Offset 号。

[0100] (4) 采集性能指标的原始数据，这些原始数据可用于两个用户侧的性能指标的计算，分别是节目启动时延，节目播放连续度。这些数据将被存储于数据库中。

[0101] 整个播放过大致程如下 ：

[0102] PlayController 向 DataScheduler 发送一个 begin 信号告诉 DataScheduler 开始对第一块媒体数据块进行调度。

[0103] DataScheduler 将调度的到媒体数据填充到 PlayBuffer 中，一旦 PlayBuffer 的数据足够一段时间的播放，DataScheduler 将给 PlayController 发送一个 ready 的信号，PlayController 到 PlayBuffer 中检测数据，并触发一系列的 VCR 操作。如果当前播放指针 P 所指向的 PlayBuffer 中的 chunk 没有数据，即标志位为 0，那么，此时就出现“卡”的现象，PlayController 向 DataScheduler 发送 Require data 的信号，DataScheduler 根据该信号所请求的 seg 去调度相应的数据块 . 当 BataScheduler 调度到该 chunk，将它填充到

PlayBuffer 中，然后给 PlayController 发一个 ready 信号，PlayController 可继续播放。如此重复下去直到用户会话结束，即用户正常播放结束或者发生停止操作。

[0104] 2，详细步骤

[0105] 为了使 VCR 操作更接近实际用户的操作行为，先进行模型的参数配置：

[0106] (1)VCR动作转移模型的概率采用文献《Providing unrestricted VCR functions in multicast video-on-demand servers.（一种在点播多媒体服务器中提供无限制 VCR 功能的方案）》In Proc. International Conf. on Multimedia Computing and Systems(ICMCS)1998, Austin TX, USA, June 1998 :66-75. 作者 Abram-Profeta E L. Shin K G. 中的通行假设，即 Pr_{play} = 0.70, Pr_{abort} = 0.05, Pr_{jf} = 0.15, Pr_{jb} = 0.10.

[0107] (2)VCR 动作状态模型中的参数采用文献《A client caching scheme for interactive video-on-demand.（一个适用于交互式点播系统的客户端缓存方案）》In Proc. IEEE of Intentional conf. on Networks(ICON)1999, Melbourne, Australia, September1999 :391-397. 作者 Branch P. Egan G. Tonkin B 和文献《Continuous-media courseware server :A study of client interactions.（连续媒体课件服务器：一项用户交互的研究）》IEEE Internet Computing, Magagine 1999,3(6) :65-73. 作者 Padhye J. Kurose J 中的通行数据，播放时间的参数是 $\mu = -4.19$, $\sigma = 1.13$，跳转持续时间的参数是 $\mu = -3.76$, $\sigma = 1.01$。

[0108] 整个节目视频播放过程可以分为数据准备和播放两个阶段，下面对这两个阶段进行详细阐述，图 6 所示的是全部两个阶段的详细过程。

[0109] 2.1，数据准备阶段

[0110] 步骤 1PlayController 模块向 DataScheduler 发送一个 begin 消息置 seg = 0，表示第一块媒体数据，DataScheduler 开始进行数据调度。PlayController 记录时间戳 TS_{begin}，即 TS_{begin} = simTime0，其中 simTime0 表示取当前仿真系统的时间戳。

[0111] 步骤 2 数据调度模块 DataScheduler 进行数据调度对 seg 进行调度，把收满的 BM 数据块转移到 PlayBuffer，同时为了最大保证播放的连续，会用自身的数据调度算法进行其它数据的调度。转到步骤 3.

[0112] 步骤 3DataScheduller 向 PlayContreller 发送 ready 信号，告知 PlayController 已经可以开始进行播放。此时，进入播放阶段。

[0113] 2.2，播放阶段

[0114] 步骤 1 是否是第一次开始播放，若是，则转到步骤 2，否则转到步骤 3。

[0115] 步骤 2 初始化 P, P_{last}, TS_0, TS_1, TS_{start}, TS_{end} 使得 P.offset = 0, P.chunk = 0, P_{last}.offset = 0, P_{last}.chunk = 0, TS_0 = -1, TS_1 = -1, TS_{start} = -1, TS_{end} = -1. 即让 P 和 P_{last} 都指向 PlayBuffer 的起始位置。

[0116] 步骤 3 由 (4) 式采用参数 ($\mu = -4.19$, $\sigma = 1.13$) 产生播放时间步长的随机数 t_0。

[0117] 步骤 4PlayController 从 PlayBuffer 根据可用 BM 个数计算播放定时器的持续时间 t，方法是：从 P 点处往后遍历 PlayBuffer，直到 BM 的 offset 不等于 -1，记录 BM 的个数 n. 若 n > 0，那么计算 n 块 BM 可以播放的时间 t，计算公式：t = n • (BMSize/BitRate)，跳到步骤 8. 若 n ≤ 0，转到步骤 5。

[0118] 步骤 5 判断若 TS_0等于 -1，则跳到步骤 7，否则跳到步骤 6。

[0119] 步骤 6 记录 TS_1= simTime0，将 TS_0，TS_1同时写入数据库，并重置 TS_0，TS_1，即 TS_0 = -1，TS_1= -1。

[0120] 步骤 7 那么 PlayController 向 DataScheduler 发送 require data 的信号，信号中携带有 seg 信息，表示出现“卡”的位置的数据块，转到数据准备阶段的步骤 2。

[0121] 步骤 8 若 t_0> t，那么令 t = t_0。

[0122] 步骤 9 启动定时器 T_{pt}，其到期时间间隔为 t. 若 TS_0等于 -1，则记录 TS_0= simTime0，若 TS_{start}等于 -1，则记录 TS_{start}= simTime0。

[0123] 步骤 10 定时器 T_{pt}到期，把指针 P 指向当前播放处，计算公式是

[0124] P.offset = P_{last}.offset+n

[0125] 播放过的时间记录到 P_{last}中，计算公式是

[0126] P_{last}.offset = P_{last}.offset+n

[0127] 步骤 11 检查是否已经播放到片尾，即 P.offset 是否等于 N，且 P.chunk 是否等于 M_{last}，若是，则跳到步骤 17，否则跳到步骤 12。

[0128] 步骤 12 按照 VCR 动作转移模型产生下一个次 VCR 操作。

[0129] 步骤 13 该 VCR 操作是否为停止，若是，则跳到步骤 17 ；否则跳到步骤 14。

[0130] 步骤 14 判断该 VCR 操作是否播放，若是，则跳到步骤 3 ；否则跳到步骤 15。

[0131] 步骤 15 对跳进和跳退操作，由 (4) 式采用参数 (μ = -3.76, σ = 1.01) 产生跳转的步长 t_{step}，其包的 chunk 个数为 n_{chunk}= t_{step}/(ChunkSize/BitRate)，包含的 BM 个数为 n_{BM}= n_{chunk}/M. 多余的 chunk 个数 r_{chunk}= n_{chunk}modM，其中 M 表示每个 BM 包含 chunk 的个数。将播放指针重置，若是跳进动作，则 P.offset = P.offset+n_{BM}，P.chunk = P.chunk+r_{chunk}；若是跳退动作，则 P.offset = P.offset-n_{BM}，P.chunk = P.chunk-r_{chunk}. 转到步骤 16。

[0132] 步骤 16 记录 TS_1= simTime0，将 TS_0，TS_1同时写入数据库，并重置 TS_0，TS_1，即 TS_0 = -1，TS_1= -1，转到步骤 3。

[0133] 步骤 17 若 TS_0不等于 -1，则记录 TS_1= simTime0，记录 TS_{end}= simTime0，将 TS_{begin}，TS_0，TS_1，TS_{start}，TS_{end}同时写入数据库。

[0134] 步骤 18 播放结束。若定时器 T_{pt}仍处在调度状态，则将之取消。

[0135] 3，性能指标计算

[0136] 仿真结束，模型采集到的性能指标的原始数据用于计算两个重要的用户侧性能指标，分别是节目启动时延和节目播放连续度。仿真过程所采集的原始数据均存储于数据库之中，可通过数据库平台进行计算。计算方法如下 ：

[0137] (1) 节目启动时延 ：T_{start}= TS_{start}-TS_{begin}，

[0138] (2) 节目播放连续度 ：由定义可知，C_{play}= t_{play}/t_{all}，在仿真过程中，假设有 n 对数据对 <TS_0，TS_1> 存储入数据库当中，则$t_{play} = \sum_{i=1}^{n}(TS_{0i} - TS_{1i})$ ，而 t_{all}= TS_{start}-TS_{end}，故有

[0139] $C_{play} = \sum_{i=1}^{n}(TS_{0i} - TS_{1i})/(TS_{start} - TS_{end})$。

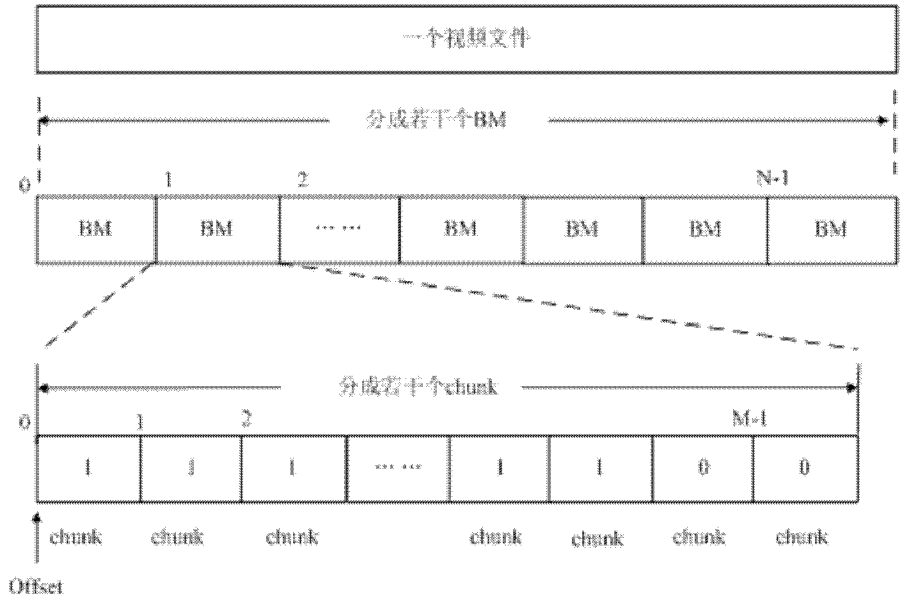

一个视频文件
分成若干个BM
0
1
2
N-1
BM
BM
… …
BM
BM
BM
BM
分成若干个chunk
0
1
2
M-1
1
1
1
… …
1
1
0
0
chunk
chunk
chunk
chunk
chunk
chunk
chunk
Offset

图 1

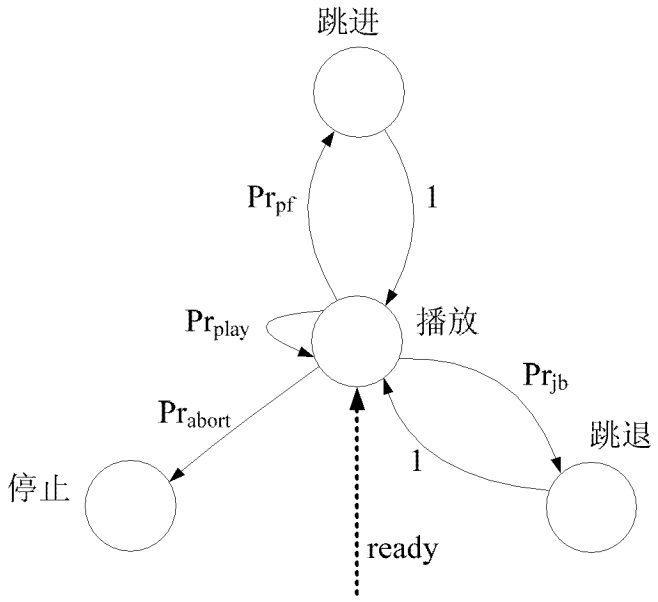

跳进
播放
停止
跳退
ready
1
1
Pr_pf
Pr_play
Pr_abort
Pr_jb

图 2

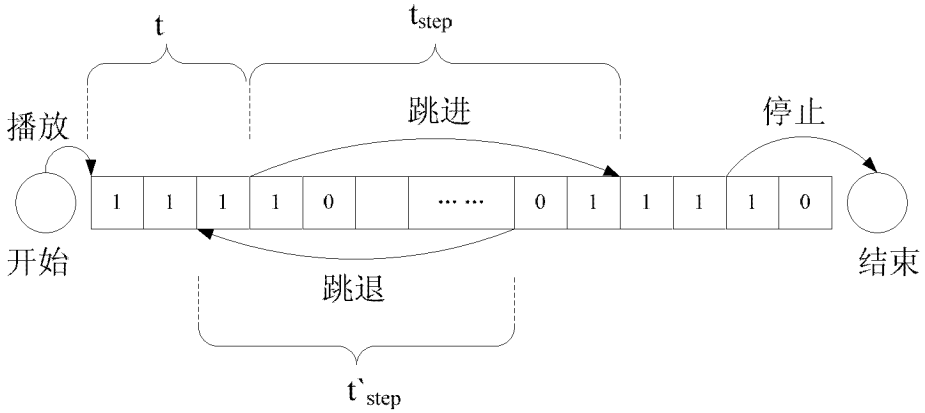

t
tstep
播放
跳进
停止
1 1 1 1 0 …… 0 1 1 1 1 0
开始
结束
跳退
t`step

图 3

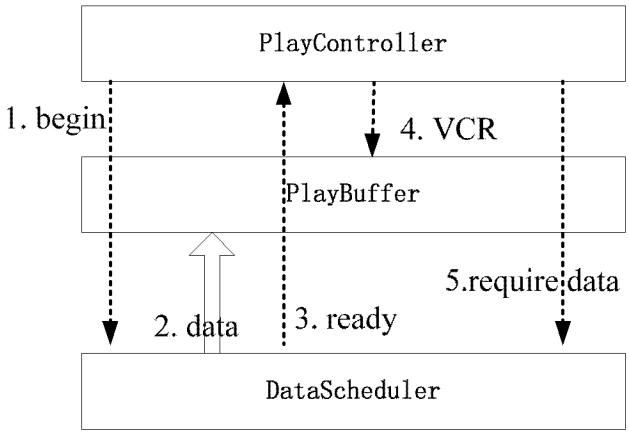

PlayController
1. begin
4. VCR
PlayBuffer
5.require data
2. data
3. ready
DataScheduler

图 4

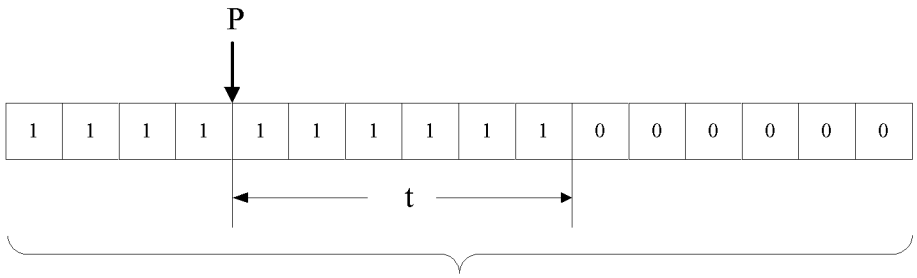

P
1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0
t

图 5

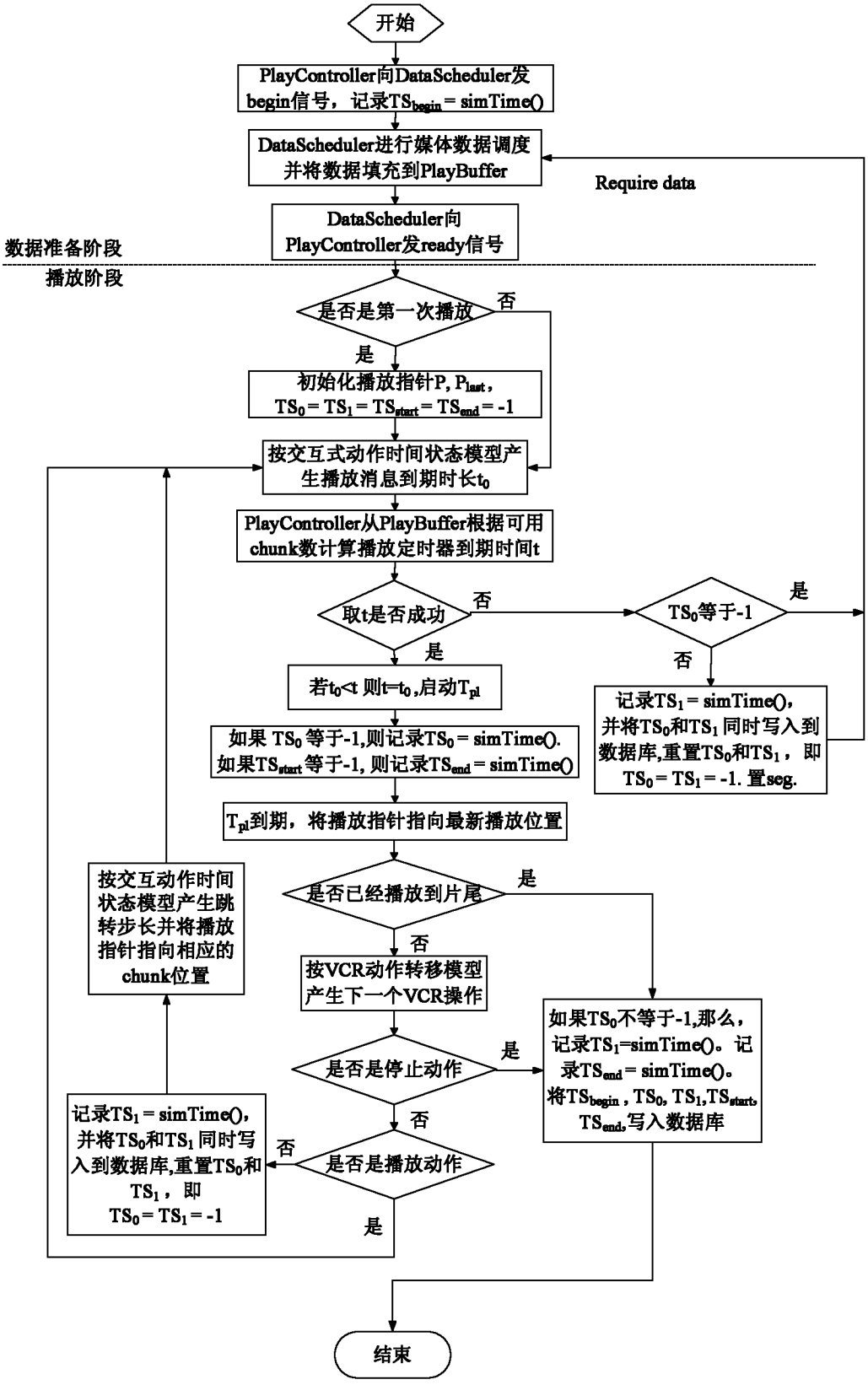

开始
PlayController向DataScheduler发begin信号，记录TS_{begin} = simTime()
DataScheduler进行媒体数据调度并将数据填充到PlayBuffer
Require data
DataScheduler向PlayController发ready信号
数据准备阶段
播放阶段
是否是第一次播放
否
是
初始化播放指针P, P_{last}，$TS_0 = TS_1 = TS_{start} = TS_{end} = -1$
按交互式动作时间状态模型产生播放消息到期时长t_0
PlayController从PlayBuffer根据可用chunk数计算播放定时器到期时间t
取t是否成功
否
是
TS_0等于-1
是
否
若t_0<t 则t=t_0,启动T_{pl}
记录TS_1 = simTime()，并将TS_0和TS_1同时写入到数据库,重置TS_0和TS_1，即$TS_0 = TS_1 = -1$. 置seg.
如果 TS_0 等于-1,则记录TS_0 = simTime(). 如果TS_{start}等于-1, 则记录TS_{end} = simTime()
T_{pl}到期，将播放指针指向最新播放位置
按交互动作时间状态模型产生跳转步长并将播放指针指向相应的chunk位置
是否已经播放到片尾
是
否
按VCR动作转移模型产生下一个VCR操作
如果TS_0不等于-1,那么，记录TS_1=simTime()。记录TS_{end} = simTime()。将TS_{begin} , TS_0, TS_1,TS_{start}, TS_{end},写入数据库
是否是停止动作
是
否
记录TS_1 = simTime()，并将TS_0和TS_1同时写入到数据库,重置TS_0和TS_1，即$TS_0 = TS_1 = -1$
否
是否是播放动作
是
结束

图 6